Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 59 (2015) 530 – 539

Procedia Computer Science

International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

Reversible Data Hiding Technique on Jpeg Image By Quad-Tree Segmentation and Histogram Shifting Method Based On Android

Rojali[a], Afan Galih Salman[b]

[a]*Mathmematics Department, School of Computer Science, Bina Nusantara Univesity,Jakarta 11480, Indonesia*

[b]*Computer Science Program, School of Computer Science, Bina Nusantara Univesity,Jakarta 11480, Indonesia*

Abstract

This research apply data hiding technique in reversible manner into Android application by means of Histogram Shifting with addition maximum and minimum point so that it expected to be capable of hiding high capacity data. The extracted image can use again Histogram Shifting reversibel Algorithm. After process of embedding secret data the resultant distorsion is relative low. Combining is carried on by scheme of hierarchically segmentation (*Quad-tree Segmentation*) which able to hierarchically partition input of the main image into several size of block based on various pixel under value of maximal capacity criterion for every block that is partitioned. These blocks is organized to be a tree structure for every block that is represented. Moreover the secret data and the partition tree information is embeded in blocks of the image where everey blocks are not squeezed. Using the methods which will be implemented to the coloured image it is expected the capacity of data insertion will increase significantly.

Peer-review under responsibility of organizing committee of the International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

Keywords: histogram shifting, quad-tree segmentation, android.

Corresponding author. Tel.: +62-21-5345830

E-mail address : rojali@binus.edu

Peer-review under responsibility of organizing committee of the International Conference on Computer Science and Computational Intelligence (ICCSCI 2015)

doi:10.1016/j.procs.2015.07.535

1. Introduction

On ths digital era, the digital content can be spread widely in the internet rapidly and easily. At the same time, copy and distribution of digital content can lead to illegal action in the public internet. The owner of copyright on digital content has been paying attention to technology of copyright protection. Some experts has provided several methods of data hiding technique to insert secret data, copyright information, or trademark into the digital content protect or secure the copyright, and only a few slight modification made on the original content. On some scheme of storing data or *watermarking*, *host media* who want to protect or to send the secret stored data, possibly meet with some distortion of the data and the data can not be transformed back into the original format on the extraction stage. No matter how, on the use of sensitive image such as medical image, military remote sensing image, and art work, it is important to guarantee that the media can be retrieved to the original shape after the stored data receiving treatment over in the case of legal considerations[7] . Therefore, a particular technique is needed, a technique which capable of hiding data without damage the storing media.

Steganography is science and art of hiding confidential message in such a way so the existence of the message can not be detected by human senses[8]. Steganography shoud be capable of retrieving the storing media to its original form as the confidential message has not been embeded yet, there are many studies concerning steganography technique which able to return the storing media to the its original form as it is required.

2. Architecture of System

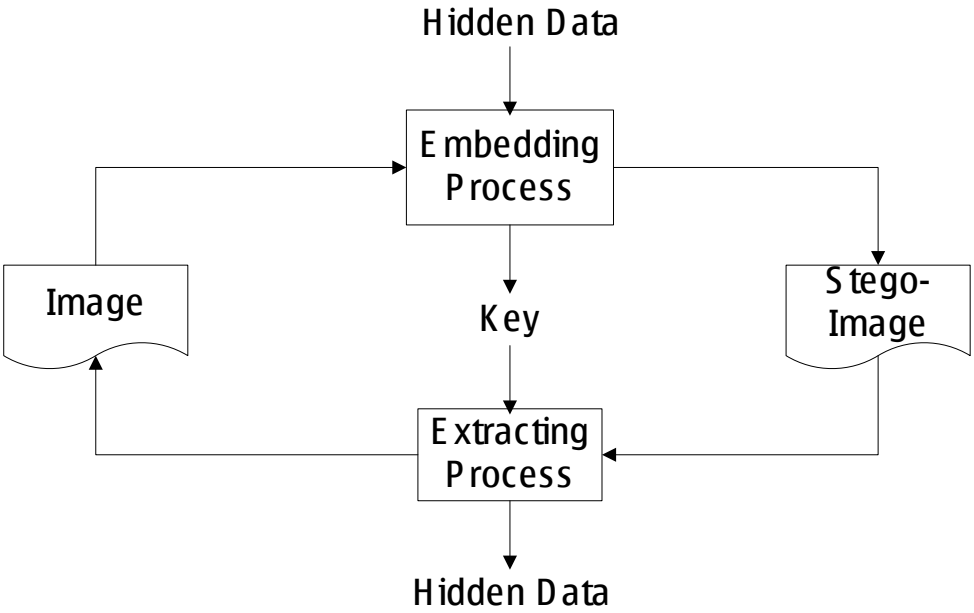

Figure 1. Circle of Steganography Process

According to consideration of image aspect that can be retrieved to its original form as before the image is inserted so we use Histogram Shifting method for data embedding. Furthermore we increase the capacity of message hiding that will add the number of maximum and minimum point to be two sets, as described in illustrated figure below.

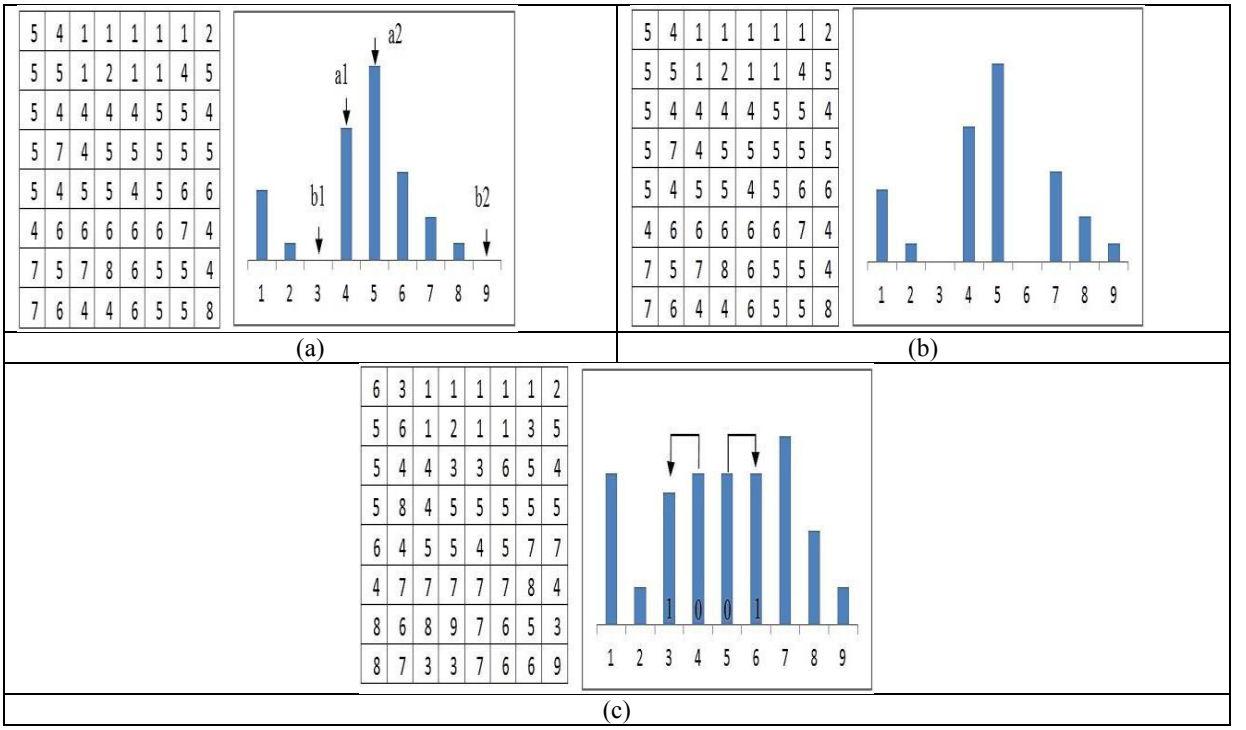

Figure 2. Flow chart of Methods Shifting with two set of maximum and minimum points, (a) The Original Image, (b) Histogram Shifting, (c) Bit Embedding

using method of Quad-tree segmentation to reduce bit of storing point minimum in the every block of histogram shifting. The more long of x and y the bigger information data that embedded. The combination of two methods can bee seen as show by the figure below.

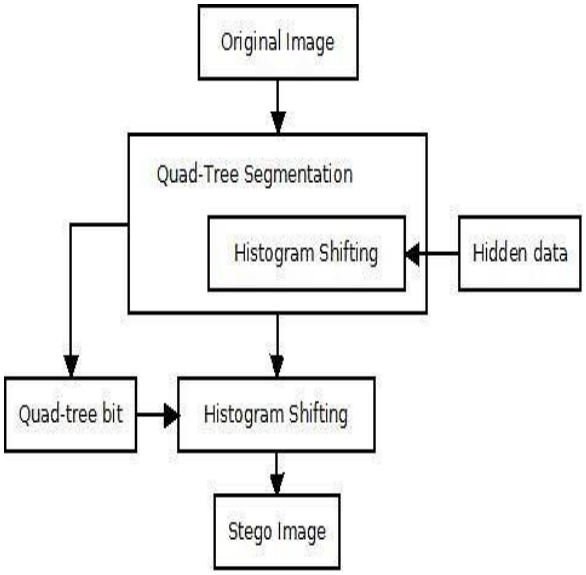

Figure 3. Embedding Process

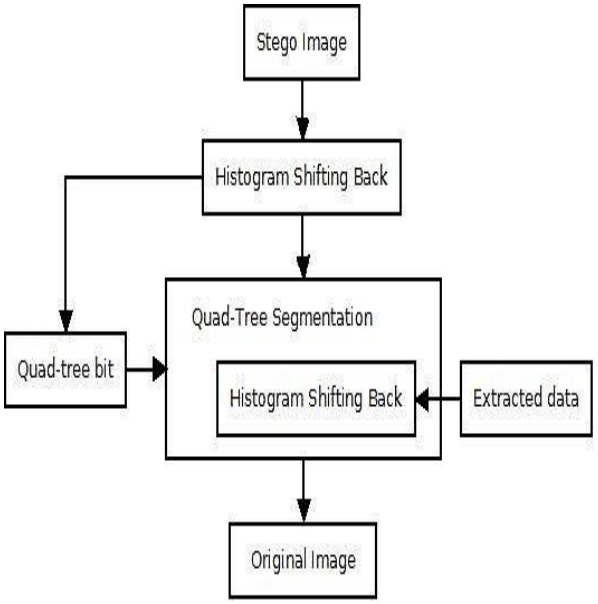

Figure 4. Extraction Process

On the implementation of this method, we use the coloured image media. The every colour canal (RGB) in image has been transformed by *Histogram Shifting*. Capacity of hiding data (pure payload) will be resulted in one block. Capaity (C) is calculated as follow:

$$C = \sum_{i=1}^{2} h_{RED}(a_i) + \sum_{i=1}^{2} h_{GREEN}(a_i) + \sum_{i=1}^{2} h_{BLUE}(a_i)$$

To connect every block of data that hidden in every colour canal (RGB) and the peak, $a_1\ dan\ a_2$, where resulted by Quad-tree segmentation and Histogram shifting lays on the whole image so it needs information that connect all of them. The connection between block and Histogram Shifting for the hidden data with *Histogram Shifting* fow bit of *Quad-Tree Segmentation* is showed in the following figure:

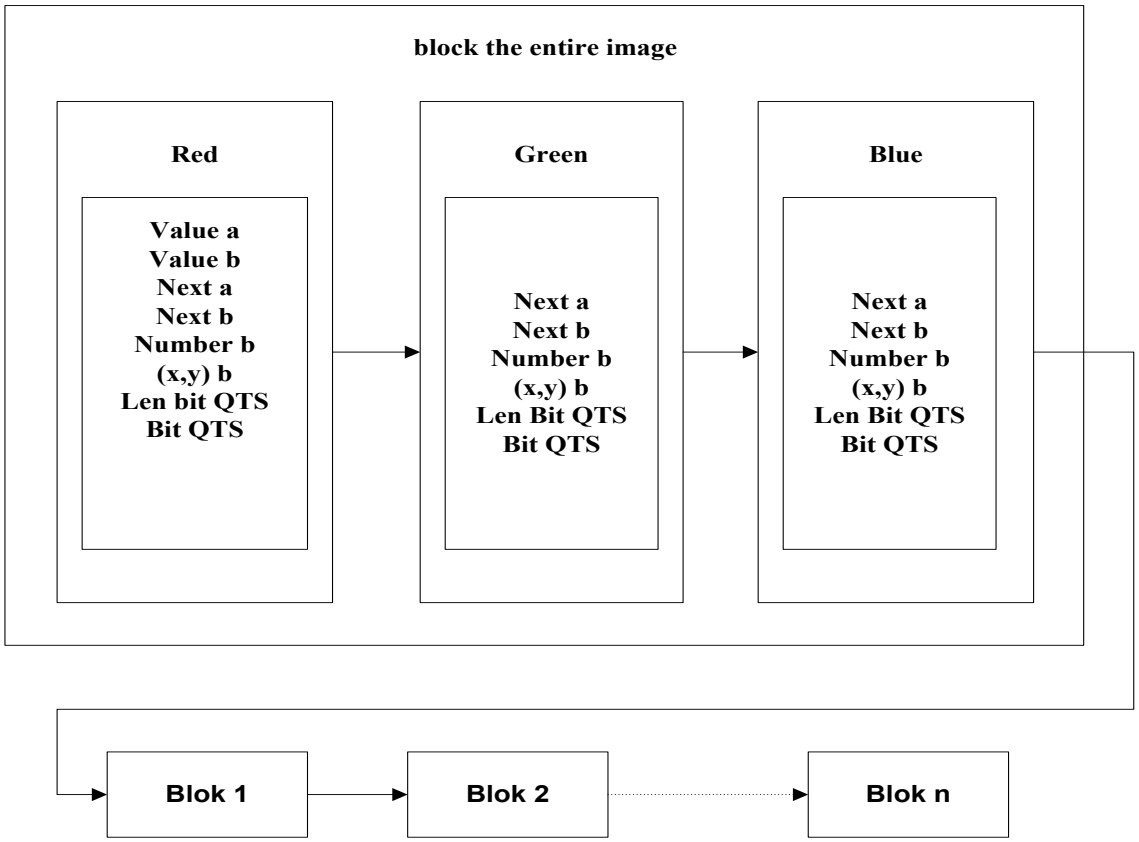

Figure 5. Connection between Overhead Information 2 and Overhead 1

According to the above figure 5. the amount of bit that is required for *next* a, *next* b, the number of b, and Len of message bit is 8 bit for each, so the total bit is 32 bits and the left is in accordance with the existing requirement.

In one block we can calculate the capacity size of the embeded data bit (*payload Size*) as follow:

$$PS(I,1) = \sum_{i=1}^{3} \left[\sum_{j=1}^{2} h_j(a_j) - (log_2 N + log_2 M) \times \sum_{j=1}^{2} h_j(b_j) - 32 \right]$$

And total of bit that can be accommodate in one image shall be as follow:

$$PS(I) = \sum_{i=1}^{total\ blok} PS(I,i)$$

The embedding and extraction processes are the develompment of the existing Histogram Shifting method, it can be seen in the following flowchart figure 6:

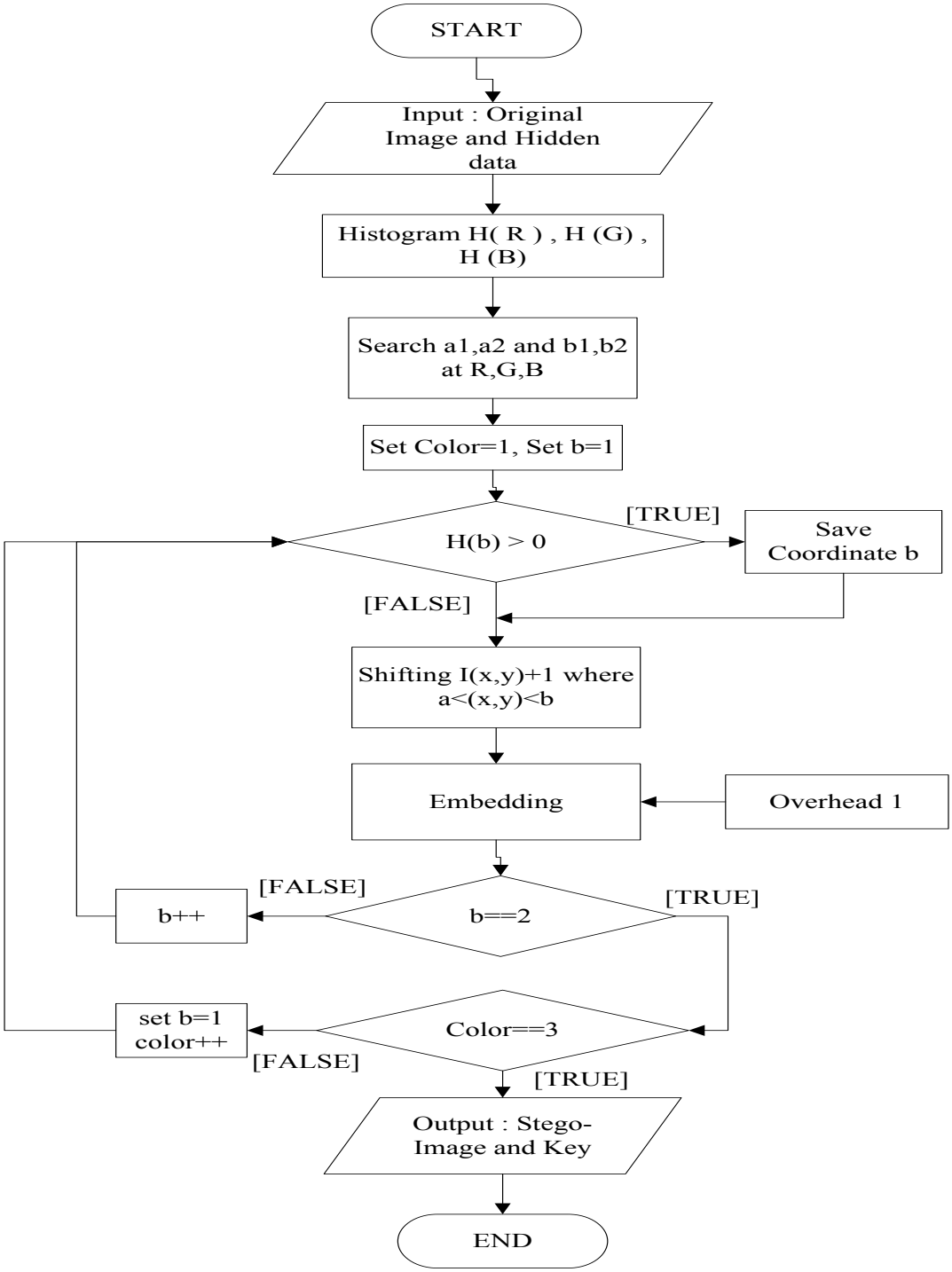

Figure 6. Flowchart of Embedding Process on Histogram Shifting.

The extraction processes are the develompment of the existing Histogram Shifting method, it can be seen in the following flowchart figure 7:

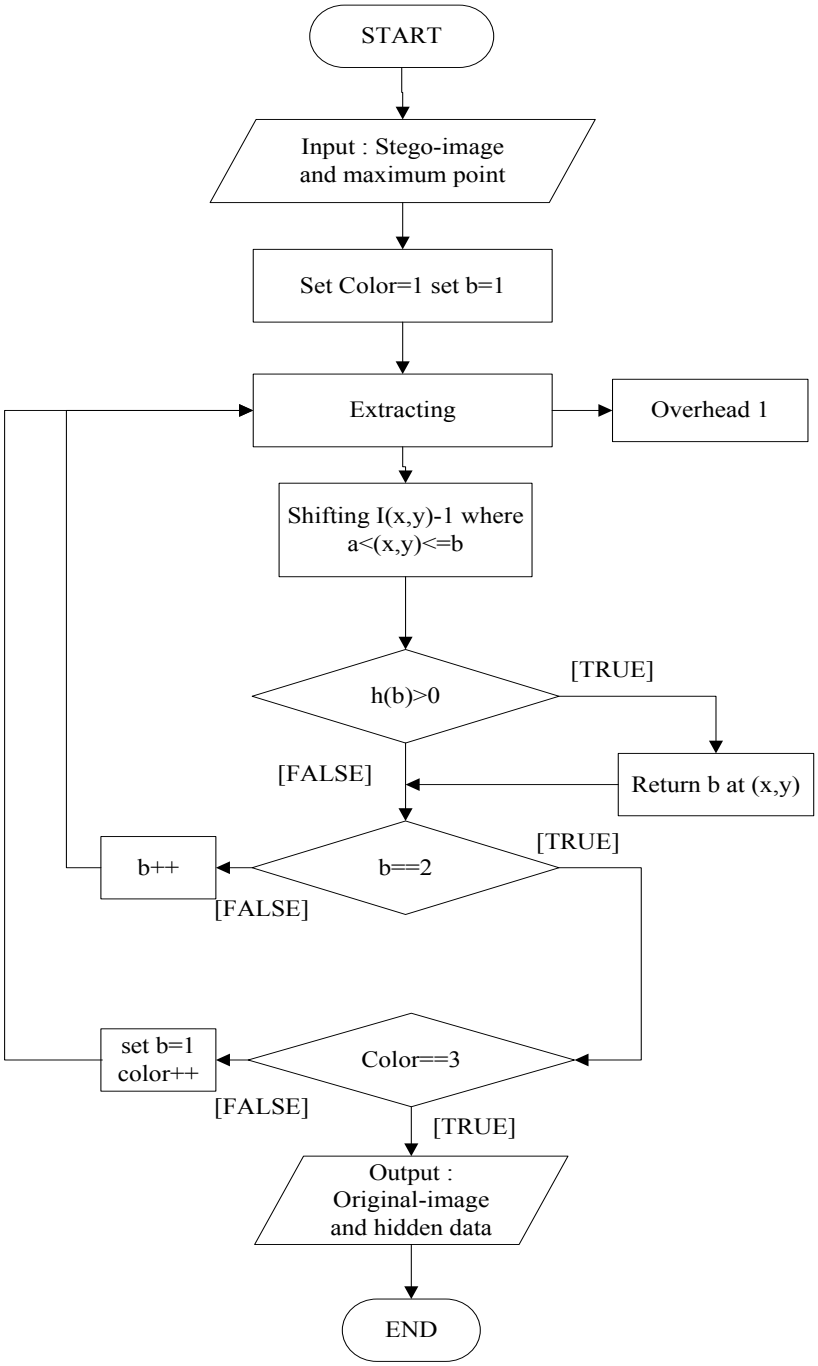

Figure 7. *Flowchart* of Extracting Process on *Histogram Shifting.*

3. Result

The message that will be embedded in the testing image can be seen as follow:

Table 1. Input Message

Type of input	*filename*	The size of bit
Message	-	128
Data	test-1.txt	1024+80
Data	test.rtf	4192+64
Data	test-2.txt	8432+80
Data	test.gif	34080+64

Before carry out embedding process, the program conduct process of Quad-tree Segmentation to the image. The result of The Quad-tree Segmentation process are bit maximum and bit structure of Quad-tree Segmentation.
After Quad-tree Segmentation process we get stucture of *Quad-tree*, then the input image will be separated or segmented.
Every segmentation of image follows figure 8. structure pattern of Quad-tree

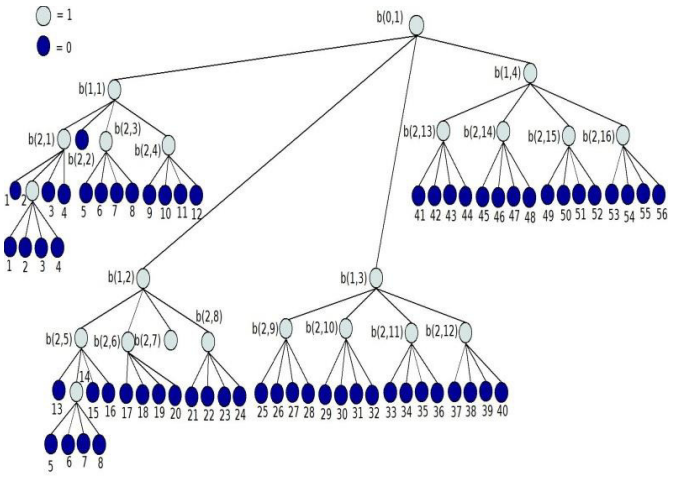

Figure 8. *Quad-tree* on Image of *Lenna*

The formation of tree structure on Quad-tree started out with rootor b (0,1) the size of *PayLoad* is calculated as follow:

$$PS(I,1) = \sum_{i=1}^{3}\left[\sum_{j=1}^{2} h(a_{ij}) - (log_2 N + log_2 M) \times \sum_{j=1}^{2} h(b_{ij}) - 33\right]$$

Example :

$Red : a_1(ha_1), b_1(hb_1), a_2(ha_2), b_2(hb_2) = 222(4021),56(0),223(4030),58(3)$

$Green : a_1(ha_1), b_1(hb_1), a_2(ha_2), b_2(hb_2) = 93(2104),1(0),96(2114),234(0)$

$Blue : a_1(ha_1), b_1(hb_1), a_2(ha_2), b_2(hb_2) = 76(3179),40(0)$

$M = 512, N = 512$

$$
\begin{aligned}
PS(I,1) &= [h(a_{R1}) - (log_2 512 + log_2 512) \times h(b_{R1}) - 33] + [h(a_{R2}) - (log_2 512 + log_2 512) \times h(b_{R2}) - 33] \\
&\quad + \cdots + [h(a_{B1}) - (log_2 512 + log_2 512) \times h(b_{B1}) - 33] \\
&\quad + [h(a_{B2}) - (log_2 512 + log_2 512) \times h(b_{B2}) - 33] \\
&= [4021 - (18) \times 0 - 33] + [4030 - (18) \times 3 - 33] + [2104 - (18) \times 0 - 33] \\
&\quad + [2114 - (18) \times 0 - 33] + [3178 - (18) \times 0 - 33] + [3218 - (18) \times 0 - 33] \\
&= 18346 \text{ bit}
\end{aligned}
$$

Moreover the result of b(0,1) is compared to *payload* total of 4 *child* that are b(1,1), b(1,2), b(1,3), and b(1,4). If the result is bigger then b(0,1) will be separated into 4 parts. Then *payload* b(1,1) is compared to *Payload* total of 4 *child* b(1,1), and so on until the *level tree* = 0.

After that for the maximum calcuation bit is taken from *Payload total in every node leaf* of *Quad-tree* Futhermore we continue on image embedding after process of *Quad-tree Segmentation* using the message listed in the Table 1, and make table of result of PSNR and process time of every message that is embedded in image as we can see below :

Table 2. The Result of PSNR and The Time Process of Embedding Message

	Message	test-1.txt	test.rtf	test-2.txt	test.gif
Lenna	58,23 dB	58,61 dB	56,24 dB	55,55 dB	50,68 dB
	(18 detik)	(18 detik)	(19 detik)	(20 detik)	(27 detik)
F-16	58,34 dB	27,82 dB	56,77 dB	55,18 dB	50,88 dB
	(20 detik)	(20 detik)	(20 detik)	(22 detik)	(30 detik)
House	59,07 dB	57,81 dB	55,63 dB	55,12 dB	52,22 dB
	(18 detik)	(18 detik)	(19 detik)	(19 detik)	(27 detik)

The calculation of MSE as follow

$$MSE = \frac{\sum_{M,N}[I_1(m,n) - I_2(m,n)]^2}{M \times N}$$

M express the long of Image and N express the width of Image, Total of square of difference in pixel of Image-*Stego*(I_1) with the Image-Original (I_2) divided by the the spacious of Image resulted in MSE.
Then the PSNR is calculated as follow:

$$PSNR = 10 \log_{10} \left(\frac{(3 \times 255)^2}{MSE} \right)$$

4. Conclusion

Based on the experiment, the produced application program is capable of embedding text message or data message into the image of the .jpg or .jpeg types. The application program is able to extract the coloured image that has been embedded at the first place. In the Histogram Shifting method the addition to maximum and minimum points is feasible. Implementation of *Quad-tree Segmentation* method on the coloured image can optimize the size of Payload. The result of the image *stego* is categorized as very good because the PSNR average produced from The stego image and original image is $\geq 40dB$. The stego image can be retrieved into the original image that is the beginning image before message embedded (*reversible*).

5. Suggestion

Some suggestions are proposed with the possibility of advance development as follow:

1. Develop n of maximum and minimum points in the Histogram Shifting method
2. Adjust the size of message requirement (*Payload*) to the Quad-tree segmentation in order to accelerate the process.

References

1. Conzalez, R. C., & Woods, R. E. (2008). *Digital Image Processing* Third Edition. Prentice Hall.
2. Cox, I. J., Miller, M. L., Bloom, J. A., Fridrich, J., & Kalker, T. (2008). *Digital Watermarking and Steganography*, Second Edition. Morgan Kaufmann.
3. Cummins, J., Diskin, P., Lau, S., & Parlett, R. (2004). *Steganography and Digital Watermarking*.
4. H, N. S. (2011). *Android*. Bandung: Informatika.
5. Hong, W., Chen, T. S., Lin, K. Y., & Chiang, W. C. (2010). A *Modified Histrogram Shifting Based Reversible Data Hiding Scheme for High Quality Images*. Information Technology Jurnal, 9, 179-183.
6. Kumar, A., & Pooja, K. (2010). *Steganography- A Data Hiding Technique*. International Journal of Computer Applications, vol. 9, no. 7, 19-23.
7. Lin, Y.-C., & Li, T.-S. (2011). Reversible *Image Data Hiding Using Quad-tree Segmentation and Histogram Shifting*. Journal of Multimedia, vol. 6, no. 4, 349-358.
8. Munir, R. (2004). *Steganografi dan Watermarking*. Departemen Teknik Informatika Institut Teknologi Bandung.
9. Ni, Z., Shi, Y.-Q., Ansari, N., & Su, W. (2006). *Reversible Data Hiding*. IEEE Transaction on Circuits and Systems for Video Technology, vol. 16, no. 3, 354-362.
10. The MathWorks, Inc. (1994-2012). *Compute peak signal-to-noise ratio (PSNR) between images*. September 1, 2012, MathWorks: http://www.mathworks.com/help/vision/ref/psnr.html
11. Tian, J. (2003). *Reversible Data Embedding Using a Difference Expansion*. IEEE Transactions on Circuits and Systems for Video Technology, vol. 13, no. 8, 890-897.